#100 DAYS OF RTL

https://github.com/Bhuvan-2602 www.LinkedIn.com/bhuvan-p-4825a9358

Description:

This design implements a 4-bit **Adder/Subtractor** using a ripple-carry structure of full adders. The operation mode is controlled by a single input cin:

- cin = 0: Perform **Addition**
- cin = 1: Perform **Subtraction**

This is achieved by using 2's complement logic: A - B = A + (~B + 1).

Objective:

Learn to design a reusable arithmetic unit that performs both addition and subtraction using basic logic components and Verilog control structures.

Inputs:

- a: 4-bit binary input
- b: 4-bit binary input
- cin: 1-bit control input (0 for Add, 1 for Subtract)

Outputs:

- result: 4-bit result of the operation
- cout: Carry out or borrow indicator

Design Approach:

- XOR input b with the cin signal to conditionally invert it.
- Set **cin** as the initial carry-in to complete 2's complement subtraction.
- Chain four full adders to form a 4-bit ripple-carry adder/subtractor.
- This allows shared hardware logic for both operations with minimal control logic.

Verilog Code:

Full Adder

```
module full_add(
    input a,
    input b,
    input cin,
    output sum,
    output cout
);
    assign sum = a ^ b ^ cin;
    assign cout = (a & b) | (b & cin) | (cin & a);
endmodule
```

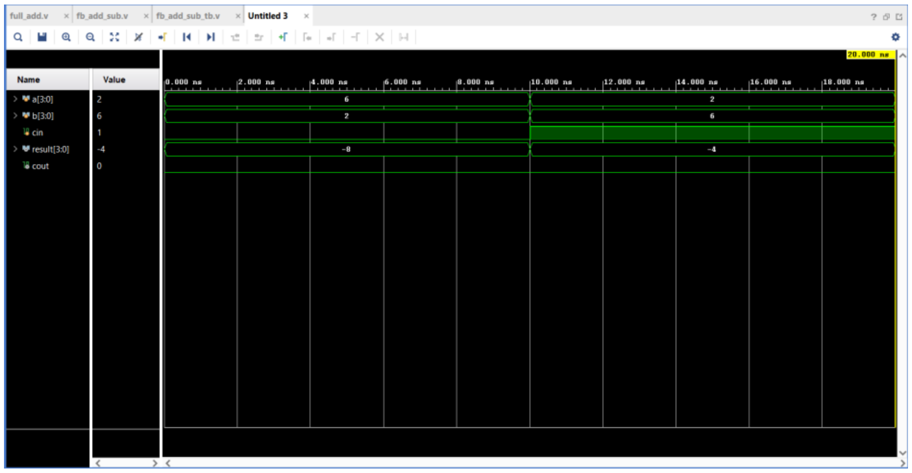
A 4-bit Adder/Subtractor

```
module fb_add_sub(
    input [3:0] a,
    input [3:0] b,
    input cin,
    output [3:0] result,
    output cout
);
    wire [3:0] b_xor;
    wire c1, c2, c3;

    assign b_xor = b ^ {4{cin}};

    full_add f1 (a[0], b_xor[0], cin, result[0], c1);
    full_add f2 (a[1], b_xor[1], c1, result[1], c2);
    full_add f3 (a[2], b_xor[2], c2, result[2], c3);
    full_add f4 (a[3], b_xor[3], c3, result[3], cout);
endmodule
```

Testbench:

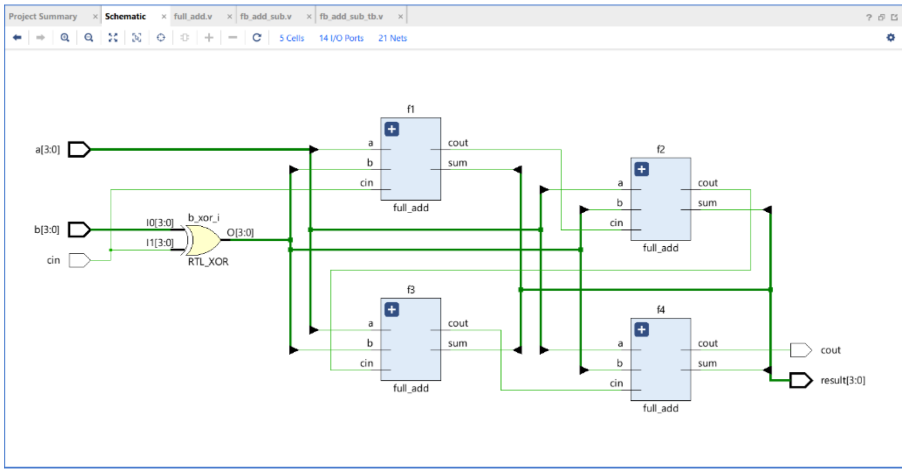
```
module fb_add_sub_tb;
    reg [3:0] a, b;
    reg cin;
    wire [3:0] result;
    wire cout;

    fb_add_sub dut (a, b, cin, result, cout);

    initial begin
        cin = 1'b0;          // Test addition: 6 + 2 = 8
        a = 4'b0110;
        b = 4'b0010;
        #10;

        cin = 1'b1;         // Test subtraction: 2 - 6 = (negative in 2's comp)
        a = 4'b0010;
        b = 4'b0110;
        #10;

        $finish();
    end
endmodule
```

Waveform:



Name	Value
a[3:0]	2
b[3:0]	6
cin	1
result[3:0]	-4
cout	0

Schematic:


Project Summary
Schematic
full_add.v
fb_add_sub.v
fb_add_sub_tb.v
5 Cells
14 I/O Ports
21 Nets
a[3:0]
b[3:0]
cin
I0[3:0]
I1[3:0]
b_xor_i
O[3:0]
RTL_XOR
f1
f2
f3
f4
a
b
cin
cout
sum
full_add
cout
result[3:0]